RECEIVED
CENTRAL FAX CENTER

FEB 08 2006

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: **Koved et al.**

Serial No.: **10/002,439**

Filed: **November 1, 2001**

For: **Method and Apparatus for Type Independent Permission Based Access Control**

§ Group Art Unit: **2131**

§ Examiner: **Abrishamkar, Kaveh**

§ Attorney Docket No.: **AUS920010941US1**

§

Certificate of Transmission Under 37 C.F.R. § 1.8(a)
I hereby certify this correspondence is being transmitted via facsimile to the Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, facsimile number (571) 273-8300, on February 8, 2006.

By: Jane M. Roberts
Jane M. Roberts

35525
PATENT TRADEMARK OFFICE
CUSTOMER NUMBER

TRANSMITTAL DOCUMENT

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:
TRANSMITTED HEREWITH:

- Appeal Brief (37 C.F.R. 41.37)

A fee of $500.00 is required for filing an Appeal Brief. Please charge this fee to IBM Corporation Deposit Account No. 09-0447. No additional fees are believed to be necessary. If, however, any additional fees are required, I authorize the Commissioner to charge these fees which may be required to IBM Corporation Deposit Account No. 09-0447. No extension of time is believed to be necessary. If, however, an extension of time is required, the extension is requested, and I authorize the Commissioner to charge any fees for this extension to IBM Corporation Deposit Account No. 09-0447.

Respectfully submitted,

Rakesh Gupta
Registration No. 57,434
AGENT FOR APPLICANTS

Duke W. Yee
Registration No. 34,285
YEE & ASSOCIATES, P.C.
P.O. Box 802333
Dallas, Texas 75380
(972) 385-8777
ATTORNEY FOR APPLICANTS

RECEIVED
CENTRAL FAX CENTER

FEB 0 8 2006

Yee & Associates, P.C.

4100 Alpha Road
Suite 1100
Dallas, Texas 75244

Main No. (972) 385-8777
Facsimile (972) 385-7766

Facsimile Cover Sheet

To: Commissioner for Patents for **Examiner Kaveh Abrishamkar** **Group Art Unit 2131**	Facsimile No.: **571/273-8300**
From: Jane M. Roberts Legal Assistant to Rakesh Garg	No. of Pages Including Cover Sheet: **40**

Message:

Transmitted herewith:

- Transmittal Document; and
- Appeal Brief.

Re: Application No.: **10/002,439**
Attorney Docket No: **AUS920010941US1**

Date: Wednesday, February 08, 2006

Please contact us at (972) 385-8777 if you do not receive all pages indicated above or experience any difficulty in receiving this facsimile.

PLEASE CONFIRM RECEIPT OF THIS TRANSMISSION BY FAXING A CONFIRMATION TO 972-385-7766.

RECEIVED
CENTRAL FAX CENTER

FEB 0 8 2006

Docket No. AUS920010941US1

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: **Koved et al.**	§	
	§	Group Art Unit: **2131**
Serial No.: **10/002,439**	§	
	§	Examiner: **Abrishamkar, Kaveh**
Filed: **November 1, 2001**	§	
	§	
For: **Method and Apparatus for Type Independent Permission Based Access Control**	§	

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Certificate of Transmission Under 37 C.F.R. § 1.8(a)
I hereby certify this correspondence is being transmitted via facsimile to the Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, facsimile number (571) 273-8300, on February 8, 2006.

By: Jane M. Roberts
Jane M. Roberts

APPEAL BRIEF (37 C.F.R. 41.37)

This brief is in furtherance of the Notice of Appeal, filed in this case on December 20, 2005.

The fees required under § 41.20(B)(2), and any required petition for extension of time for filing this brief and fees therefore, are dealt with in the accompanying TRANSMITTAL OF APPEAL BRIEF.

02/10/2006 EFLORES 00000069 10002439
01 FC:1402 500.00 DA

REAL PARTY IN INTEREST

The real party in interest in this appeal is the following party: International Business Machines.

RELATED APPEALS AND INTERFERENCES

With respect to other appeals or interferences that will directly affect, or be directly affected by, or have a bearing on the Board's decision in the pending appeal, no such appeals or interferences are present.

RECEIVED
CENTRAL FAX CENTER
FEB 0 8 2006

STATUS OF CLAIMS

A. TOTAL NUMBER OF CLAIMS IN APPLICATION

Claims in the application are: 1-30

B. STATUS OF ALL THE CLAIMS IN APPLICATION

1. Claims canceled: None
2. Claims withdrawn from consideration but not canceled: None
3. Claims pending: 1-30
4. Claims allowed: None
5. Claims rejected: 1-30
6. Claims objected to: None

C. CLAIMS ON APPEAL

The claims on appeal are: 1-30

STATUS OF AMENDMENTS

No amendments were filed after the final office action dated September 22, 2005.

SUMMARY OF CLAIMED SUBJECT MATTER

A. CLAIM 1 - INDEPENDENT

The subject matter of claim 1 is directed to a method of controlling access to computer system resources based on permissions. The claimed invention comprises the steps of receiving a request for access to a computer system resource (figure 6; Specification, p. 6, ll. 21-23, p. 15, ll. 8-11, p. 19, ll. 26-31); determining if a superclass permission of a required permission is present in each protection domain of an access control context (figure 6; figure 9, reference numeral 970; figure 10, reference numeral 1030; Specification, p. 20, ll. 16-21), where the superclass permission is a super class of the required permission (Specification, p. 18, ll. 5-32; Specification, p. 21, ll. 11-14; figures 5, 7A-7C); adding the required permission to a permission collection if the superclass permission of the required permission is present in each protection domain of the access control context (Specification, p. 20, ll. 16-23; figure 9, reference numeral 980; figure 10, reference numeral 1040); and granting access to the resource if the superclass permission of the required permission is present in each protection domain of the access control context (figure 6; figure 9, reference numeral 995; figure 10, reference numeral 1050; Specification, p. 24, ll. 4-13, p. 25, ll. 18-22).

B. CLAIM 3 – DEPENDENT

The subject matter of claim 3 is directed to the method of claim 1 and further comprising the steps of determining the required permission based on a CodeSource associated with the request (Specification, p. 19, ll. 26-31; figure 9, reference numeral 920); and determining the superclass permission of the required permission based on the required permission (Specification, p. 20, ll. 16-21).

C. CLAIM 4 – DEPENDENT

The subject matter of claim 4 is directed to the method of claim 1. In this claim, determining if a superclass permission of a required permission is present in each protection

domain includes determining if at least one permission collection in each protection domain includes the superclass permission (Specification, p. 17, ll. 24-27, p. 23, ll. 29-32; figure 9, reference numeral 970).

D. CLAIM 5 – DEPENDENT

The subject matter of claim 5 is directed to the method of claim 1. In this claim, the adding of the required permission to a permission collection includes creating a new permission collection and adding the required permission to the new permission collection (Specification p. 20, ll. 12-23).

E. CLAIM 6 – DEPENDENT

The subject matter of claim 6 is directed to the method of claim 5. In this claim, the adding of the required permission to a permission collection also includes adding any subclass permissions of the required permission to the new permission collection (Specification, p. 20, l. 31 - p. 21, l. 10).

F. CLAIM 8 – DEPENDENT

The subject matter of claim 8 is directed to the method of claim. In this claim, the adding of the required permission to a permission collection includes adding the permission to a permission collection associated with the superclass permission (Specification, p. 20, ll. 12-23).

G. CLAIM 11 - INDEPENDENT

The subject matter of claim 11 is directed to a computer program product in a computer readable medium for controlling access to computer system resources based on permissions. The claimed invention comprises instructions for receiving a request for access to a computer system resource (figure 6; Specification, p. 6, ll. 21-23, p. 15, ll. 8-11, p. 19, ll. 26-31); instructions for determining if a superclass permission of a required permission is present in each protection domain of an access control context (figure 6; figure 9, reference numeral 970; figure 10,

reference numeral 1030; Specification, p. 20, ll. 16-21), where the superclass permission is a super class of the required permission (Specification, p. 18, ll. 5-32; Specification, p. 21, ll. 11-14; figures 5, 7A-7C); instructions for adding the required permission to a permission collection if the superclass permission of the required permission is present in each protection domain of the access control context (Specification, p. 20, ll. 16-23; figure 9, reference numeral 980; figure 10, reference numeral 1040); and instructions for granting access to the computer system resource if the superclass permission of the required permission is present in each protection domain of the access control context (figure 6; figure 9, reference numeral 995; figure 10, reference numeral 1050; Specification, p. 24, ll. 4-13, p. 25, ll. 18-22).

H. CLAIM 13 – DEPENDENT

The subject matter of claim 13 is directed to the computer program product of claim 11. This claim includes additional instructions for determining the required permission based on a CodeSource associated with the request (Specification, p. 19, ll. 26-31; figure 9, reference numeral 920); and additional instructions for determining the superclass permission of the required permission based on the required permission (Specification, p. 20, ll. 16-21).

I. CLAIM 14 – DEPENDENT

The subject matter of claim 14 is directed to the computer program product of claim 11. In this claim the instructions for determining if a superclass permission of a required permission is present in each protection domain include instructions for determining if at least one permission collection in each protection domain includes the superclass permission (Specification, p. 17, ll. 24-27, p. 23, ll. 29-32; figure 9, reference numeral 970).

J. CLAIM 15 – DEPENDENT

The subject matter of claim 15 is directed to the computer program product of claim 11. In this claim, the instructions for adding the required permission to a permission collection include instructions for creating a new permission collection and instructions for adding the required permission to the new permission collection (Specification p. 20, ll. 12-23).

K. CLAIM 16 – DEPENDENT

The subject matter of claim 16 is directed to the computer program product of claim 15. In this claim, the instructions for adding the required permission to a permission collection additionally include instructions for adding any subclass permissions of the required permission to the new permission collection (Specification, p. 20, l. 31 - p. 21, l. 10).

L. CLAIM 18 – DEPENDENT

The subject matter of claim 18 is directed to the computer program product of claim 11. In this claim, the instructions for adding the required permission to a permission collection include instructions for adding the permission to a permission collection associated with the superclass permission (Specification p. 20, ll. 12-23).

M. CLAIM 21 - INDEPENDENT

The subject matter of claim 21 is directed to an apparatus for controlling access to computer system resources based on permissions. The claimed invention comprises means for receiving a request for access to a computer system resource (figure 6; Specification, p. 6, ll. 21-23, p. 15, ll. 8-11, p. 19, ll. 26-31); means for determining if a superclass permission of a required permission is present in each protection domain of an access control context (figure 6; figure 9, reference numeral 970; figure 10, reference numeral 1030; Specification, p. 20, ll. 16-21), wherein the superclass permission is a super class of the required permission (Specification, p. 18, ll. 5-32; Specification, p. 21, ll. 11-14; figures 5, 7A-7C); means for adding the required permission to a permission collection if the superclass permission of the required permission is present in each protection domain of the access control context (Specification, p. 20, ll. 16-23; figure 9, reference numeral 980; figure 10, reference numeral 1040); and means for granting access to the computer system resource if the superclass permission of the required permission is present in each protection domain of the access control context (figure 6; figure 9, reference numeral 995; figure 10, reference numeral 1050; Specification, p. 24, ll. 4-13, p. 25, ll. 18-22).

N. CLAIM 23 – DEPENDENT

The subject matter of claim 23 is directed to the apparatus of claim 21. This claim contains additional means for determining the required permission based on a CodeSource associated with the request (Specification, p. 19, ll. 26-31; figure 9, reference numeral 920); and additional means for determining the superclass permission of the required permission based on the required permission (Specification, p. 20, ll. 16-21).

O. CLAIM 24 – DEPENDENT

The subject matter of claim 24 is directed to the apparatus of claim 21. In this claim, the means for determining if a superclass permission of a required permission is present in each protection domain includes means for determining if at least one permission collection in each protection domain includes the superclass permission (Specification, p. 17, ll. 24-27, p. 23, ll. 29-32; figure 9, reference numeral 970).

P. CLAIM 25 – DEPENDENT

The subject matter of claim 25 is directed to the apparatus of claim 21. In this claim, the means for adding the required permission to a permission collection includes means for creating a new permission collection and means for adding the required permission to the new permission collection (Specification p. 20, ll. 12-23).

Q. CLAIM 26 – DEPENDENT

The subject matter of claim 26 is directed to the apparatus of claim 25. In this claim, the means for adding the required permission to a permission collection also includes adding any subclass permissions of the required permission to the new permission collection (Specification, p. 20, l. 31 - p. 21, l. 10).

R. CLAIM 27 - DEPENDENT

The subject matter of claim 27 is directed to the apparatus of claim 21. This claim additionally contains means for retrieving the access control context for a thread of execution that sent the request for access to the computer system resource (Specification, p. 20, ll. 1-6; figure 6, reference numeral 640; figure 8; figure 9, reference numeral 940).

S. CLAIM 28 - DEPENDENT

The subject matter of claim 28 is directed to the apparatus of claim 21. In this claim, the means for adding the required permission to a permission collection includes means for adding the permission to a permission collection associated with the superclass permission (Specification p. 20, ll. 12-23).

T. CLAIM 29 - DEPENDENT

The subject matter of claim 29 is directed to the apparatus of claim 21. In this claim, the means for determining if a superclass permission of a required permission is present in each protection domain of an access control context (Specification, p. 20, ll. 7-11), and the means for adding the required permission to a permission collection if the superclass permission of the required permission is present in each protection domain of an access control context operate based on a method called by the required permission in response to an "implies" method operating on the required permission (Specification, p. 21, l. 30 – p. 22, l. 21, p. 23, l. 29 – p. 24, l. 13; figure 9, reference numerals 970, 975, 980, 985, 990, 995).

U. CLAIM 30 - DEPENDENT

The subject matter of claim 30 is directed to the apparatus of claim 23. In this claim, the means for determining the required permission based on a CodeSource associated with the request and means for determining the superclass permission of the required permission based on the required permission operate based on a security policy file (Specification, p. 24, ll. 14-30; figure 10, reference numerals 1010, 1020, 1030).

GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

A. GROUND OF REJECTION 1 (Claims 1-30)

Claims 1-30 stand rejected under 35 U.S.C. § 102(b) as anticipated by *Gong* et al., Typed, Parameterized, and Extensible Access Control Permissions, United States Patent 6,047,377 (hereinafter, "*Gong*").

ARGUMENTS

A. The Reference Does Not Anticipate Claims 1-2, 4, 7, 9-12, 14, 17, 19- 22, 24, 27, and 29-30

Claim 1 is a representative claim in this group and reads as follows:

> 1. A method of controlling access to computer system resources based on permissions, comprising:
> receiving a request for access to a computer system resource;
> determining if a superclass permission of a required permission is present in each protection domain of an access control context, wherein the superclass permission is a super class of the required permission;
> adding the required permission to a permission collection if the superclass permission of the required permission is present in each protection domain of the access control context; and
> granting access to the resource if the superclass permission of the required permission is present in each protection domain of the access control context.

The Examiner has rejected this claim stating:

> Regarding claim 1, Gong discloses:
> A method of controlling access to computer system resources based on permissions, comprising:
> "receiving a request for access to a computer system resource" (Figure 7 item 750, column 6 lines 36-46, column 18 line 29 — column 19 line 36);
> "determining if a superclass permission of a required permission is present in each protection domain of an access control context, wherein the superclass permission is a super class of the required permission" (column 6 lines 36-46, column 18 lines 29-45);
> "adding the required permission to a permission collection if the superclass permission of the required permission is present in each protection domain of the access control context (column 17 lines 1-5, column 19 lines 37-43); and "granting access to the resource if the superclass permission of the required permission is present in each protection domain of the access control context' (column 10 lines 59-67, column 19 lines 4-36).

Final office action dated September 22, 2005, pp. 5-6, (emphasis removed).

A prior art reference anticipates the claimed invention under 35 U.S.C. § 102 only if every element of a claimed invention is identically shown in that single reference, arranged as they are in the claims. *In re Bond*, 910 F.2d 831, 832, 15 U.S.P.Q.2d 1566, 1567 (Fed. Cir.

1990). All limitations of the claimed invention must be considered when determining patentability. *In re Lowry*, 32 F.3d 1579, 1582, 32 U.S.P.Q.2d 1031, 1034 (Fed. Cir. 1994). Anticipation focuses on whether a claim reads on the product or process a prior art reference discloses, not on what the reference broadly teaches. *Kalman v. Kimberly-Clark Corp.*, 713 F.2d 760, 218 U.S.P.Q. 781 (Fed. Cir. 1983). In this case each and every feature of the presently claimed invention in claim 1 is not identically shown in the cited reference, arranged as they are in the claims.

More specifically, contrary to Examiner's assertion, *Gong* does not teach the step of "determining if a *superclass permission* of a required permission is present in each protected domain of an access control context, wherein the superclass permission is a *super class* of the required permission", as recited in claim 1. The Examiner cites the following sections from *Gong* as teaching this claimed feature:

> When the validation method is invoked for a particular permission object belonging to a permission subclass, the validation method indicates whether a given permission is encompassed by the permission represented by the particular permission object. For example, the validation method of a permission object PO1 may be invoked to determine whether the permission represented by another permission object PO2 is encompassed in the permission represented by PO1, where both PO1 and PO2 belong to classes that descend from said permission super class.
>
> In step 754, a request is received for a determination of whether the action is authorized. The determination is based on the permission required to perform the action. In this example, the resource manager invokes an access controller to determine whether the permission required is authorized for the entity requesting access. The access controller receives the request and a required permission object which was transmitted by the resource manager.
>
> In step 754, the validation methods of one or more permission objects is invoked in order to determine whether an action is authorized based on the permission required. An action is authorized if every protection domain object associated with an object requesting a determination of whether an action is authorized contains a permission represented by a permission object that encompasses the required permission for the action.

Gong, col. 6, ll. 36-46; col. 18, l. 29-45.

Gong uses the term permission super class in the first cited section, but all of the relevant operations described in *Gong* are with respect to encompassed permissions and permissions

encompassing, but not with respect to a super class. The term super class permission is defined in *Gong* as follows:

> Because all permissions share some common attributes, it is useful and efficient to represent all categories of permissions with one class that is a super class of all permission classes. The super class for all permission classes is herein referred to as the permission super class. Each permission class is a subclass of the permission super class.
>
> The permission super class contains fields which represent attributes common to all permissions. One such field is an action field, which represents the action attribute common to all permissions.
>
> In addition to sharing attributes, the permission super class establishes a set of common methods that are useful for and inherited by all permission objects, such as a get.sub.-- action method. The common action method returns a value representing the action field. An implementation for the get.sub.-- action method may be provided in the permission super class. According to one embodiment, the implementation for the get.sub.-- action method simply returns the value of the action field.

Gong, col. 8, ll. 19-38.

As can be seen, *Gong* defines super class in terms of commonly known object oriented hierarchy of classes. In a typical superclass-subclass hierarchy, the methods and attributes of a super class may be inherited and modified by a sub-class that derives from the super class. However, no teaching is present to support the Examiner's interpretation of *Gong* in which a "permission represented by a permission object that encompasses the required permission" is equivalent to *Gong's* permission super class, or a Superclass permission as recited in claim 1.

A permission encompassing the required permission simply means a larger permission that "encompasses" or inherently includes a smaller permission, but not a "superclass" permission in the hierarchy of permissions classes. *Gong's* disclosure supports this logical construction in the following section:

> When the validation method is invoked for a particular permission object belonging to a permission subclass, the validation method indicates whether a given permission is encompassed by the permission represented by the particular permission object. For example, the validation method of a permission object PO1 may be invoked to determine whether the permission represented by another permission object PO2 is encompassed in the permission represented by PO1, where both PO1 and PO2 belong to classes that descend from said permission super class. In this we can determine whether a permission to perform a first action, represented by

> PO1, authorizes a request to perform a second action, which requires a second permission represented by PO2, by invoking the validation method of PO1 to determine whether the first permission represents an authorization to perform the requested second action.
>
> According to another aspect of the invention, permissions represent actions on targets. Thus, a first permission object can specify a first action and a first target, and a second permission object can specify a second action and a second target. A determination is made of the whether the permission represented by the first permission object encompasses the permission represented by the second permission object based on whether the first action implies the second action and the first target implies the second target. In this we can determine, for example, whether a permission to perform a first action on a first target, represented by the first permission object, authorizes a request to perform a second action on second target, which requires a second permission represented by the second object, by determining whether first action implies the second action, and the first target implies the second target.

Gong, col. 3, ll. 3-35

In a superclass-subclass hierarchy, the attributes and methods of the superclass can be defined in mutually exclusive ways in one or more child subclasses. This property of a superclass entity is different from *Gong's* notion of an encompassing entity. For example, a permission P1 to withdraw $300 encompasses a permission P2 to withdraw $200. However, this does not mean that a parent permission P that provides for only a number and an activity associated with that number "encompasses" the child permission number representing $300 and the action representing the withdrawal thereof, within the context of *Gong*. This example shows inheritance, but not "encompassing" as *Gong* describes the term. *Gong* specifically uses the term "encompass" and not superclass to indicate this difference, as is evident from the following section in *Gong*:

> One permission can imply another. When one permission implies another permission, that one permission is said to encompass the other permission. For example, a permission to write to a directory, such "c:/", can imply a permission to write to any file in the directory, such as "c:/thisfile". Furthermore, an attribute of a permission can imply an attribute of another permission. For example, in some implementations, the action attribute of a permission to "write" implies an action attribute of a permission to "read". An amount attribute of a permission to withdraw three hundred dollars implies another attribute of a permission to withdraw two hundred dollars.

> Usually, a permission encompasses another permission when all the permission attributes of one permission imply all the corresponding permission attributes of another permission. For example, a permission to "write" to file "d:/somefile" implies a permission to "read" from file "d:/somefile" because a "write" implies a "read". However, a permission to "write" to file "d:/somefile" does not imply a permission to "read" from file "d:/otherfile" because "d:/somefile" does not imply "d:/otherfile".

Gong, col. 7, ll. 30-50

Gong teaches that when a first permission "implies" a second permission, the first permission encompasses the second permission. This express statement in *Gong* describes what *Gong* means by permission encompassing the required permission. The statement makes clear that *Gong* does not mean a permission super class, or superclass permission as defined in Appellants' specification, to be a permission encompassing the required permission. (Specification p. 18, ll. 6-17). *Gong* further clarifies the distinction between permission super class and permission encompassing the required permission within the context of *Gong's* invention in the section defining permission super class quoted below.

> Because all permissions share some common attributes, it is useful and efficient to represent all categories of permissions with one class that is a super class of all permission classes. The super class for all permission classes is herein referred to as the permission super class. Each permission class is a subclass of the permission super class.
>
> The permission super class contains fields which represent attributes common to all permissions. One such field is an action field, which represents the action attribute common to all permissions.
>
> In addition to sharing attributes, the permission super class establishes a set of common methods that are useful for and inherited by all permission objects, such as a get.sub.-- action method. The common action method returns a value representing the action field. An implementation for the get.sub.-- action method may be provided in the permission super class. According to one embodiment, the implementation for the get.sub.-- action method simply returns the value of the action field.

Gong, col. 8, ll. 19-38.

Gong's description is careful in describing the permission encompassing the required permission separately and distinctly from the permission super class, not leaving room for an interpretation that substitutes one term for another. The Examiner, however, interprets the two terms – "permission super class" and "permission encompassing the required permission" – as

interchangeable. *Gong* does not support Examiner's interpretation. *Gong's* "permission encompassing the required permission" does not mean "permission superclass" as claimed by Appellants. Therefore, contrary to Examiner's assertion, *Gong* does not teach the claimed feature, "determining if a *superclass permission* of a required permission is present in each protected domain of an access control context, wherein the superclass permission is a *super class* of the required permission." Instead, *Gong* only teaches operations with respect to encompassing permissions. Thus, *Gong* does not teach determining if a superclass permission of a required permission is present in each protection domain of an access control context, wherein the superclass permission is a super class of the required permission as recited in claim 1.

In addition, the Examiner incorrectly asserts that *Gong* teaches another feature of claim 1, "adding the required permission to a permission collection if the superclass permission of the required permission is present in each protection domain of the access control context." In support of the Examiner's assertion, the Examiner cites to the following section of *Gong*:

> If a protection domain is already associated with the code source, the domain mapper adds a mapping of the new class and protection domain to a mapping of classes and protection domains maintained by the domain mapper 448.
>
> ... Typed permissions facilitate the establishment of new permissions. When a new category of permissions is desired, a new subclass is created. The particular rules or policy that govern whether the permissions granted a principal are encompassed by permission in the new category are implemented in the validation method of the new subclass representing permissions in the new subclass.

Gong, col. 17, ll. 1-5, col. 19, ll. 37-43

Examiner's assertion with respect to this claim feature again suffers from the improper interchanging of the terms "permission encompassing the required permission" and "superclass permission" as illustrated above. The sections from *Gong* cited by the Examiner do not cure this defect in the Examiner's argument but rather re-emphasize it. Cited sections from *Gong* actually maintain the consistency of separation of terms "permission encompassing the required permission" and "superclass permission", lending further support to the distinction between, and non-interchangeability of, the two terms.

Therefore, contrary to Examiner's assertion, *Gong* does not teach the claimed feature, "adding the required permission to a permission collection if the *superclass* permission of the required permission is present in each protection domain of the access control context."

In addition, the Examiner incorrectly asserts that *Gong* teaches "granting access to the resource if the superclass permission of the required permission is present in each protection domain of the access control context" in claim 1. In support of the Examiner's assertion, the Examiner cites the following portion of *Gong*:

> Next, the code in the implementation 242 ensures that each attribute of the permission represented by the object reference is implied by each attribute of the permission represented by the object whose validation method is invoked. For example, if the action field and the target field of the permission object referred to by the object reference are identical to the action field and target field of the permission object whose validation method is invoked, then the validation method returns a true Boolean value.
>
> ... In this example, the access controller first determines the protection domain associated with the first object represented on call stack 610, which is object a. The protection domain associated with object a is protection domain I. The validation method of the first permission object, permission object 282 (in FIG. 6), is invoked, passing in the required permission object as a parameter. As mentioned earlier, the required permission represented by the required permission object is a permission to "disenable" "channel-5". When the validation method of the first permission object is invoked the validation method indicates that the required permission is not encompassed. Next, the validation method of permission object 286 (in FIG. 6) is invoked. The invocation of the validation method of permission object 286 indicates that the required permission is encompassed.
>
> The access controller then invokes the validation methods of the permission objects in the next protection domain object, protection domain object J, in the manner described. Each invocation of the validation methods of permission object 622 and permission object 626 indicates that the required permission is not encompassed.
>
> At step 764, a determination is made of whether the action requested was authorized. If every associated protection domain contains a permission object that represents a permission encompassing the required permission, then the requested action is authorized. When the requested action is authorized, control passes to step 768, where the action is performed before execution of the steps ceases. In this example, because not every

> protection domain object contained a permission encompassing the required permission, performance of the steps ends. The requested action is not executed.

Gong, col. 10, ll. 59-67, col. 19, ll. 4-36.

Examiner's assertion with respect to this claim feature again suffers from the improper interchanging of the terms "permission encompassing the required permission" and "superclass permission" as illustrated above. The sections from *Gong* cited by the Examiner do not cure this defect in the Examiner's argument but rather re-emphasize it. Cited sections from *Gong* actually maintain the consistency of separation of terms "permission encompassing the required permission" and "superclass permission," thereby lending further support to the distinction between, and non-interchangeability of, the two terms.

Therefore, contrary to Examiner's assertion, *Gong* does not teach the claimed feature, "granting access to the resource if the *superclass* permission of the required permission is present in each protection domain of the access control context."

As a result *Gong* does not teach at least three features of claim 1, which is a representative claim of this group. Consequently, *Gong* does not anticipate claims 1-2, 4, 7, 9-12, 14, 17, 19- 22, 24, 27, and 29-30 under 35 U.S.C. § 102(b).

B. <u>The Reference Does Not Teach All the Features of Claims 3, 13, and 23</u>

Claim 3 is a representative claim in this group of claims and reads as follows:

> 3. The method of claim 1, further comprising:
> determining the required permission based on a CodeSource associated with the request; and
> determining the superclass permission of the required permission based on the required permission.

The Examiner has rejected this particular claim stating:

> Claim 3 is rejected as applied above in rejecting claim 1. Furthermore, Gong discloses:
> The method of claim 1, further comprising:
> "determining the required permission based on a CodeSource associated with the request' (column 14 lines 28-36, column 15 lines 65-67); and

> "determining the superclass permission of the required permission based on the required permission" (column 6 lines 36-46, column 18 lines 29-45).

Final office action dated September 22, 2005, pp. 6, (emphasis removed).

Contrary to the Examiner's assertion, *Gong* does not teach the "determining the superclass permission of the required permission based on the required permission" as recited in claim 3.

Gong, in the cited section, discloses:

> When the validation method is invoked for a particular permission object belonging to a permission subclass, the validation method indicates whether a given permission is encompassed by the permission represented by the particular permission object. For example, the validation method of a permission object PO1 may be invoked to determine whether the permission represented by another permission object PO2 is encompassed in the permission represented by PO1, where both PO1 and PO2 belong to classes that descend from said permission super class.
>
> ... In step 754, a request is received for a determination of whether the action is authorized. The determination is based on the permission required to perform the action. In this example, the resource manager invokes an access controller to determine whether the permission required is authorized for the entity requesting access. The access controller receives the request and a required permission object which was transmitted by the resource manager.
>
> In step 754, the validation methods of one or more permission objects is invoked in order to determine whether an action is authorized based on the permission required. An action is authorized if every protection domain object associated with an object requesting a determination of whether an action is authorized contains a permission represented by a permission object that encompasses the required permission for the action.

Gong, col. 6, ll. 36-46, col. 18, ll. 29-45

Gong figure 7 is reproduced in § A, *supra*.

As before, the Examiner's assertion with respect to the claim feature "determining the superclass permission of the required permission based on the required permission" suffers from the improper interchanging of the terms "permission encompassing the required permission" and "superclass permission" as illustrated above. The sections from *Gong* cited by the Examiner do not cure this defect in the Examiner's argument but rather re-emphasize it. Cited sections from *Gong* actually maintain the consistency of separation of terms "permission encompassing the

required permission" and "superclass permission," thereby lending further support to the distinction between, and non-interchangeability of, the two terms.

Assuming, arguendo, that the Examiner's assertion that *Gong's* permission that encompasses the required permission is the same as Appellants' superclass permission is correct, *Gong* still does not teach the claimed feature, "determining the superclass permission of the required permission based on the required permission" in claim 3 for yet another reason. The cited sections from *Gong* teach determining if the required permission is encompassed by the permission represented by the particular permission object being validated against. The operation that determines a true or false logic is vastly different from the operation that determines an object. *Gong* teaches the former type of operation, and claim 3 is directed towards the latter. Determining a superclass permission of the required permission as the claimed feature does, is in effect determining an object that is an object-oriented class. An object-oriented class is not a Boolean value. *Gong* teaches determining whether a permission encompassing the required permission is present – an inquiry that results only in a Boolean true or false answer.

In other words, Gong teaches whether something is present, while the presently claimed invention in claim 3 recites determining what something is, such as what is the required permission and what is the super class permission. Thus, the process taught by Gong in the cited sections is an entirely different process as recited in claim 3.

Furthermore, *Gong's* entire disclosure fails to teach the actual determination of the permission super class of the required permission in the manner of claim 3. Therefore, *Gong* teaches a step of nature different from the nature of the feature claimed. Therefore, *Gong* does not teach all the features of claim 3. Consequently, *Gong* does not anticipate claim 3.

Therefore, contrary to Examiner's assertion, *Gong* does not teach the claimed feature, "determining the *superclass* permission of the required permission based on the required permission."

As a result, *Gong* does not teach all the features of claim 3. Additionally, claim 3 depends from claim 1, which *Gong* also does not anticipate. Consequently, *Gong* does not anticipate claims 3, 13, and 23 under 35 U.S.C. § 102(b).

C. **The Reference Does Not Teach All the Features of Claims 5, 15, and 25**

The Examiner incorrectly applies *Gong's* teaching and rejects as nonpersuasive, the Appellants' arguments advanced in response to prior office action rejecting claim 5.

Claim 5 is a representative claim in this group of claims and reads as follows:

> 5. The method of claim 1, wherein adding the required permission to a permission collection includes creating a new permission collection and adding the required permission to the new permission collection.

The Examiner has rejected Appellants' prior arguments distinguishing this claim from *Gong*, asserting,

> Regarding claim 5-6, the applicant argues that the CPA does not teach "creating anew [sic] permission collection and adding the required permission to the new permission collection" and "includes adding any subclass permissions of the required permission to the new permission collection." This argument is not found persuasive. The CPA discloses "when a new category of permissions is desired, a new subclass is created" (column 19 lines 36-38), which is creating a new permission collection. Furthermore, the CPA teaches "the particular rules or policy that govern whether the permissions granted a principal are encompassed by permission in the new category are implemented in the validation method of the new subclass representing permissions in the new subclass" (column 19 lines 38-43), which includes all the subset permissions according to the superclass permission.

Final office action dated September 22, 2005, pp. 3-4.

The Examiner is mistaking the creation of a new permission subclass that *Gong* teaches, with the creation of a new permission collection that is used in claim 5. As described in the specification as shown below, a "permission collection" is a grouping of permissions defined by a developer:

> A permission collection is a grouping of permissions defined by a developer. Every time a developer defines a new permission, the developer must also define a permission collection to which the new permission belongs or assume the default implementation. These permission collections may be assigned to classes 450 of the applet 420. Thus, when the classes are instantiated, the JVM looks at the permission collections assigned to the classes and generates a protection domain based on the assigned permission collections.

Specification p. 14, ll. 6-15.

Even assuming, *arguendo*, that the permission collection in claim 5 is comparable to any of *Gong's* entities, the permission collection in claim 5 may only be equivalent to *Gong's*

Protection Domain Object class and/or subclasses, but not the Permission subclass. Therefore, contrary to Examiner's assertion, creating a new *permission subclass* in *Gong* does not teach the feature "creating a new *permission collection* and adding the required permission to the new permission collection" of claim 5.

Furthermore, the citation to column 19, lines 38-43 of the reference is irrelevant to the claimed features of claim 5. The section cited from *Gong* states:

> In this example, because not every protection domain object contained a permission encompassing the required permission, performance of the steps ends. The requested action is not executed.
>
> Typed permissions facilitate the establishment of new permissions. When a new category of permissions is desired, a new subclass is created. The particular rules or policy that govern whether the permissions granted a principal are encompassed by permission in the new category are implemented in the validation method of the new subclass representing permissions in the new subclass.

Gong, col. 19, ll. 38-43.

As can be seen, the cited *Gong* section pertains to the implementation of the policies that govern whether the permissions granted a principal are encompassed by permission in the new category in the reference embodiment. Teachings from this section of *Gong* have no bearing on the features of claim 5 in which adding the required permission to a permission collection includes creating a new permission collection and adding the required permission to the new permission collection.

Therefore, the Examiner has rejected claims 5, 15, and 25 based on an incorrect understanding of the claimed invention and the reference.

D. <u>The Reference Does Not Teach All the Features of Claims 6, 16, and 26</u>

Claim 6 is representative of claims 16 and 26, and reads as follows:

> 6. The method of claim 5, wherein adding the required permission to a permission collection further includes adding any subclass permissions of the required permission to the new permission collection.

Because claim 6 depends from claim 5, the same distinctions between *Gong* and the invention of claim 5 can be made for claim 6 as well.

The Examiner further rejects claim 6 as being anticipated by *Gong* citing a different section.

The Examiner has rejected this particular claim stating:

> Claim 6 is rejected as applied above in rejecting claim 1. Furthermore, Gong discloses:
> The method of claim 5, wherein adding the required permission to a permission collection further includes "adding any subclass permissions of the required permission to the new permission collection" (column 16 line 56— column 17 line 13).

Final office action dated September 22, 2005, p. 7, (emphasis removed).

Gong, in the cited section, discloses:

> The domain mapper object 448 contains a mapping between classes and protection domains objects. Protection domain objects 482 contain a set of permissions. Protection domain objects are associated with the permission objects they contain, and with the classes to which a protection domain object is mapped to by domain mapper object 448.
>
> Protection domain objects 482 are created when new classes are received by code executor 410. When a new class is received, domain mapper 448 determines whether a protection domain is already associated with the code source. The domain mapper maintains data indicating which protection domains have been created and the code sources associated with the protection domains. If a protection domain is already associated with the code source, the domain mapper adds a mapping of the new class and protection domain to a mapping of classes and protection domains maintained by the domain mapper 448.
>
> If a protection domain object is not associated with the code source of the new class, a new protection domain object is created and populated with permissions. The protection domain is populated with those permission that are mapped to the code source of the new class based on the mapping of code sources to permissions in the policy object. Finally, the domain mapper adds a mapping of the new class and protection domain to the mapping of classes and protection domains as previously described.

Gong, col. 16, l. 56 - col. 17, l. 13

The cited section from *Gong* fails to suggest the claim feature "adding any *subclass* permissions of the required permission to the new permission collection." *Gong* adds only the permission that is being requested but does not teach or suggest adding subclasses of the

permission being requested to the permission domain object. The *Gong* disclosure as a whole also does not teach or suggest adding permission subclasses to the permission domain object. Therefore, *Gong* does not teach the features of claim 6.

As a result, *Gong* does not teach all the features of claim 6. Consequently, *Gong* does not anticipate claims 6, 16 and 26 under 35 U.S.C. § 102(b).

E. The Reference Does Not Teach All the Features of Claims 8, 18, and 28

Claim 8 is a representative claim in this group and reads as follows:

> 8. The method of claim 1, wherein adding the required permission to a permission collection includes adding the permission to a permission collection associated with the superclass permission.

The Examiner has rejected this particular claim stating:

> Claim 8 is rejected as applied above in rejecting claim 1. Furthermore, Gong discloses:
> The method of claim 1, wherein adding the required permission to a permission collection includes "adding the permission to a permission collection associated with the superclass permission" (column 16 line 56— column 17 line 13).

Final office action dated September 22, 2005, p. 7, (emphasis removed).

Gong sections cited by the Examiner have been reproduced in § C, *supra*.

The cited section of *Gong* fails to teach the claim feature "adding the permission to a permission collection associated with the superclass permission." *Gong* teaches adding only the permission being requested to the permission collection but no association is present between the permission collection and the superclass permission as recited in claim 8. Furthermore, this association is not suggested anywhere in the entire *Gong* disclosure. Therefore, *Gong* does not teach or suggest that the permission collection *is associated with the superclass permission*.

In addition, even if the Examiner's assertion that *Gong's* permission that encompasses the required permission is the same as Appellants' superclass permission could be considered correct for argument's sake, *Gong* does not teach "adding the permission to a permission collection associated with the superclass permission" as recited in claim 8 for another reason. The cited section from *Gong* teaches adding the required permission to the new or existing protection domain object associated with the *code source of the new class* associated with the requested

permission. Only for the sake of clarity in drawing the distinction, *Gong's* protection domain object will be considered an equivalent of permission collection feature of Appellants' claims.

Appellants do not concede that such is the case, but will refer to both objects as permissions collection in the following analysis.

Gong teaches adding permissions to a permissions collection based on the code source of the new class associated with the requested permission, and not based on the super class of the requested permission. The two methods of adding the permission to permissions collection are distinct from each other. The code-source-method for addition is necessarily implemented differently from the superclass-permission-method in software programming. A method for adding permissions to permissions collection encoded in the first way will not be structurally identical, nor work logically the same, for the second. The two methods involve different entities to test upon and therefore cannot be substituted one for the other in order to add required permission to permissions collection.

Gong discloses another method for adding permissions to permissions collection. In the relevant section quoted below, *Gong* states:

> Another method, add.sub.-- permission, adds a permission object to the set of permission objects contained in the PermissionCollection object. The add.sub.-- permission method has a parameter of the type Permission. In the case of a homogenous PermissionClass object, for example, the method to add a permission object does not add a permission object if it is not of the same type (i.e. class) as any other permission object already contained in the PermissionCollection object. The method returns a Boolean flag indicating whether or not a permission object was added.

Gong, col. 12, ll. 15-24.

The disclosure in this section also does not teach adding permission to permissions collection based on the superclass permission of the required permission. This method, the only other method disclosed in *Gong* for adding permissions to permissions collection, tests whether the permission to be added is homogeneous with the other permissions existing within the permissions collection. In other words, this method adds a permission to a permissions collection only when the permission being added is of the same class as the other permissions in that permissions collection. Again, the code for testing for a match between the class of the required permission and the classes of the existing permissions, does not look or function the same as the code to test for association of the permissions collection with the superclass

permission of the required permission. Teaching the former test, as *Gong* does, will not enable a person of ordinary skill in the art to make and use the invention based on the later test, as Appellants do.

As a result, *Gong* does not teach all the features of claim 8. Therefore, *Gong* does not anticipate claim 8 under 35 U.S.C. § 102(b).

CONCLUSION

For the foregoing reasons, *Gong* does not anticipate claims 1-30 under 35 U.S.C. § 102(b). Therefore, Appellants respectfully urge that the Board of Appeals reverse the rejection of claims 1-30 under 35 U.S.C. § 102(b) and allow the claims.

Rakesh Garg
Reg. No. 57,434
YEE & ASSOCIATES, P.C.
PO Box 802333
Dallas, TX 75380
(972) 385-8777

CLAIMS APPENDIX

The text of the claims involved in the appeal reads:

1. A method of controlling access to computer system resources based on permissions, comprising:

receiving a request for access to a computer system resource;

determining if a superclass permission of a required permission is present in each protection domain of an access control context, wherein the superclass permission is a super class of the required permission;

adding the required permission to a permission collection if the superclass permission of the required permission is present in each protection domain of the access control context; and

granting access to the resource if the superclass permission of the required permission is present in each protection domain of the access control context.

2. The method of claim 1, wherein the request is received from bytecode.

3. The method of claim 1, further comprising:

determining the required permission based on a CodeSource associated with the request; and

determining the superclass permission of the required permission based on the required permission.

4. The method of claim 1, wherein determining if a superclass permission of a required permission is present in each protection domain includes determining if at least one permission collection in each protection domain includes the superclass permission.

5. The method of claim 1, wherein adding the required permission to a permission collection includes creating a new permission collection and adding the required permission to the new permission collection.

6. The method of claim 5, wherein adding the required permission to a permission collection further includes adding any subclass permissions of the required permission to the new permission collection.

7. The method of claim 1, further comprising retrieving the access control context for a thread of execution that sent the request for access to the computer system resource.

8. The method of claim 1, wherein adding the required permission to a permission collection includes adding the permission to a permission collection associated with the superclass permission.

9. The method of claim 1, wherein the steps of determining if a superclass permission of a required permission is present in each protection domain of an access control context, and adding the required permission to a permission collection if the superclass permission of the required permission is present in each protection domain of an access control context are performed by a

method called by the required permission in response to an implies method operating on the required permission.

10. The method of claim 3, wherein the steps of determining the required permission based on a CodeSource associated with the request and determining the superclass permission of the required permission based on the required permission are performed based on a security policy file.

11. A computer program product in a computer readable medium for controlling access to computer system resources based on permissions, comprising:

first instructions for receiving a request for access to a computer system resource;

second instructions for determining if a superclass permission of a required permission is present in each protection domain of an access control context, wherein the superclass permission is a super class of the required permission;

third instructions for adding the required permission to a permission collection if the superclass permission of the required permission is present in each protection domain of the access control context; and

fourth instructions for granting access to the computer system resource if the superclass permission of the required permission is present in each protection domain of the access control context.

12. The computer program product of claim 11, wherein the request is received from bytecode.

13. The computer program product of claim 11, further comprising:

fifth instructions for determining the required permission based on a CodeSource associated with the request; and

sixth instructions for determining the superclass permission of the required permission based on the required permission.

14. The computer program product of claim 11, wherein the second instructions for determining if a superclass permission of a required permission is present in each protection domain include instructions for determining if at least one permission collection in each protection domain includes the superclass permission.

15. The computer program product of claim 11, wherein the third instructions for adding the required permission to a permission collection include instructions for creating a new permission collection and instructions for adding the required permission to the new permission collection.

16. The computer program product of claim 15, wherein the third instructions for adding the required permission to a permission collection further include instructions for adding any subclass permissions of the required permission to the new permission collection.

17. The computer program product of claim 11, further comprising fifth instructions for retrieving the access control context for a thread of execution that sent the request for access to the computer system resource.

18. The computer program product of claim 11, wherein the third instructions for adding the required permission to a permission collection include instructions for adding the permission to a permission collection associated with the superclass permission.

19. The computer program product of claim 11, wherein the second and third instructions are part of a method called by the required permission in response to an implies method operating on the required permission.

20. The computer program product of claim 13, wherein the fifth and sixth instructions are executed based on a security policy file.

21. An apparatus for controlling access to computer system resources based on permissions, comprising:

means for receiving a request for access to a computer system resource;

means for determining if a superclass permission of a required permission is present in each protection domain of an access control context, wherein the superclass permission is a super class of the required permission;

means for adding the required permission to a permission collection if the superclass permission of the required permission is present in each protection domain of the access control context; and

means for granting access to the computer system resource if the superclass permission of the required permission is present in each protection domain of the access control context.

22. The apparatus of claim 21, wherein the request is received from bytecode.

23. The apparatus of claim 21, further comprising:

means for determining the required permission based on a CodeSource associated with the request; and

means for determining the superclass permission of the required permission based on the required permission.

24. The apparatus of claim 21, wherein the means for determining if a superclass permission of a required permission is present in each protection domain includes means for determining if at least one permission collection in each protection domain includes the superclass permission.

25. The apparatus of claim 21, wherein the means for adding the required permission to a permission collection includes means for creating a new permission collection and means for adding the required permission to the new permission collection.

26. The apparatus of claim 25, wherein the means for adding the required permission to a permission collection further includes adding any subclass permissions of the required permission to the new permission collection.

27. The apparatus of claim 21, further comprising means for retrieving the access control context for a thread of execution that sent the request for access to the computer system resource.

28. The apparatus of claim 21, wherein the means for adding the required permission to a permission collection includes means for adding the permission to a permission collection associated with the superclass permission.

29. The apparatus of claim 21, wherein the means for determining if a superclass permission of a required permission is present in each protection domain of an access control context, and the means for adding the required permission to a permission collection if the superclass permission of the required permission is present in each protection domain of an access control context operate based on a method called by the required permission in response to an implies method operating on the required permission.

30. The apparatus of claim 23, wherein the means for determining the required permission based on a CodeSource associated with the request and means for determining the superclass permission of the required permission based on the required permission operate based on a security policy file.

EVIDENCE APPENDIX

No evidence needs to be presented.

RELATED PROCEEDINGS APPENDIX

No related proceedings are pending.